

WARNING! The following entry contains registry related code. Incorrectly modifying the registry could render your system unusable, requiring reinstallation. If you are unsure as to what the following code is doing or are not comfortable working with the registry, navigate away from this page and visit the [MSDN Registry Reference](#) first.

USE THIS CODE ONLY AT YOUR OWN RISK!! Neither the administrators of this wiki, nor anyone associated with the development and production of Liberty BASIC, recommends, supports, or will engage in the correction of or reparation for any harm or damage caused by anyone's experimentation with code designed to read, write, or affect the Windows Registry in any way.

Retrieve XP Product Key

This program queries the registry for the product key used to install Windows XP on the current machine and displays it onscreen.

In previous versions of Windows, the key used at installation time was stored in string form within the registry and was easily viewable using RegEdit. Currently, that 25 character key is encoded in a 15 byte chunk of a 164 byte (I believe) registry entry called DigitalProductID. Having once found myself in a position of needing to get a 20 user office fully license-compliant in a very short amount of time, I failed to record which key was used on which machine. When one workstation eventually suffered a complete hard disk failure, I was left not knowing which of those 20 keys I could use to install XP to a new hard disk on that workstation. Some online searching led to an algorithm in VBScript to decode the installation key from the DigitalProductID entry (contained within the DecodeKey sub below). This is a simplified version of the resulting program I wrote to extract the key from every machine in our office network. This version displays only the local machine's product key in a notice window. It has been used on Windows XP Pro SP2 and Server 2003 Enterprise SP1.

```
'XPKey 1.0
'Cassio Ferreira
'Jan 2006
'If you make a zillion dollars utilizing
'this code, please donate 1% of that to
'a non-profit charity of your choice.
```

```
nomainwin
```

```
cr$ = chr$(13)
```

```
'Set up and load array to hold
'all 24 possible key characters.
dim kCh$(23)
data "B","C","D","F","G","H",_
    "J","K","M","P","Q","R",_
```

```
    "T","V","W","X","Y","2",_
    "3","4","6","7","8","9"
for x = 0 to 23
    read chars$
    kCh$(x) = chars$
next x
restore

'Set up structs necessary for the
'registry calls. In this instance,
'one to receive a handle to the open
'registry key, and one to receive
'the size of the data, respectively.
struct H,          key          as ulong
struct cbData,    size          as long

'Set up variables to hold the subkey
'within HKEY_LOCAL_MACHINE and the
'name of the entry within that subkey,
'respectively.
subKey$ = "SOFTWARE\Microsoft\Windows NT\CurrentVersion"
value$ = "DigitalProductID"

'Open "advapi32.dll" to make the query
open "advapi32.dll" for dll as #aa32
aa32.isOpen = 1

'Make the call to open the subkey
'indicated in "subKey$", specifying
'KEY_READ access rights.
callDll #aa32, "RegOpenKeyExA",_
    _HKEY_LOCAL_MACHINE as ulong,_
    subKey$              as ptr,_
    0                    as ulong,_
    _KEY_READ            as ulong,_
    H                    as struct,_
    errorReturned        as long

'If the key was opened successfully...
if not(errorReturned) then

    'Extract the handle from the struct
    hKey = H.key.struct

    'Query the entry once, specifying 0
    'for the data buffer pointer, which
```

```
'places the size of the data into
'the struct.
callDll #aa32, "RegQueryValueExA", _
    hKey          as ulong, _
    value$        as ptr, _
    0             as ulong, _
    0             as ulong, _
    0             as ulong, _
    cbData        as struct, _
    errorReturned as long

'If the query was successful...
if not(errorReturned) then

    'Space-fill a variable big enough
    'to hold the entry data.
    dataLen = cbData.size.struct
    buff$ = space$(dataLen)

    'Query the entry again, this time
    'passing it the buffer variable.
    callDll #aa32, "RegQueryValueExA", _
        hKey          as ulong, _
        value$        as ptr, _
        0             as ulong, _
        0             as ulong, _
        buff$         as ptr, _
        cbData        as struct, _
        errorReturned as long

    'If that was successful...
    if not(errorReturned) then

        'Extract just the relevant bytes
        'and send it off to decode.
        buff$ = mid$(buff$, 53, 15)

        call DecodeKey buff$, xpKey$

        'Proudly display it
        notice "XPKey 1.0" + cr$ + _
            "Windows XP Product key: " + _
            cr$ + cr$ + xpKey$
    else
        notice "Error" + cr$ + "Error retrieving key data"
    end if
```

```
    else
        notice "Error" + cr$ + "Error querying data size"
    end if
else
    notice "Error" + cr$ + "Unable to open key"
end if

close #aa32
aa32.isOpen = 0

[exit]
    if (aa32.isOpen) then close #aa32
end

sub DecodeKey buffer$, byref keyString$
    redim binVals(14)
    for x = 1 to 15
        binVals(x - 1) = asc(mid$(buffer$, x, 1))
    next x

    keyString$ = ""
    for i = 24 to 0 step -1
        c = 0
        for k = 14 to 0 step -1
            c = (c * 256) xor binVals(k)
            binVals(k) = int(c / 24)
            c = c mod 24
        next k
        keyString$ = kCh$(c) + keyString$
        if (not(i mod 5) and not(not(i))) then keyString$ = ("-" +
keyString$)
    next i
end sub
```